## REMARKS

Claim 1 is objected to as not detailing a tangible structure performing the method for sharing a class. However, method claims never recite tangible structure. Therefore, the provisions of M.P.E.P. § 2106 have no application to a method claim.

As pointed out in M.P.E.P. § 2106 at section 2100-18, a claim process must be limited to a practical application of the abstract idea or mathematical algorithm in the technological arts. Claim 1 is a practical application since it allows applications to share a class. From the examples of claimed statutory processes, as set forth on page 2100-18, it is clear that a process that has any practical application is statutory. Here, the invention is applicable to enabling applications to use shared memory.

The claim cannot be implemented by writing on a piece of paper since the claim requires running applications - something a piece of paper cannot do.

Therefore, reconsideration is respectfully requested.

The cited reference is asserted to teach running two applications and enabling the applications to share a class. However, there is no assertion that it teaches duplicating the member for each application "in a shared memory." There is no discussion of where the asserted member data is stored. Nothing can be found in the reference. Therefore, the reference does not teach "duplicating the member data for said class for each application <u>in a shared memory</u>."

Moreover, the reference does not teach providing a handle "to each application" to enable "each application" to access its member data "in the shared memory." Again, the "in the shared memory" is the same limitation discussed above.

However, there is no provision of a handle to each application so that each application can access its member data, assuming the Examiner is correct that the static field correspond to member data.

Instead, the cited reference uses a totally different technique. He uses the second generated class Counter$sMethods. It contains a table mapping application identifiers into per-application copies of Counter$sFields. See right column on 356 at the middle of the page. Counter$sFields has access methods that look up a copy of the Counter$sFields corresponding to the current application and accesses the named field. Thus, rather than providing handles to the application, a method is used to simply access the asserted member data. Thus, it appears that the applications themselves are not able to access the member data, are not enabled to access the member data in shared memory, and they are provided no handle to do so.

It is suggested on page 4 of the office action that it is well known in the art that a handle is any token that a program can use to identify and access an object such as a device, a file, a method, or a dialogue box. It is asserted that the cited reference teaches the application IDs used to access the correct static fields. But, even if that is true, the application ID is never provided "to each application" and it is never provided "to enable each application to access its member data." Instead, the ID is exclusively used by the methods copy Counter$sFields.

Thus, the cited reference is inadequate to make out a *prima facie* rejection.

Therefore, claim 1, its dependent claims, claim 9, its dependent claims, and claim 17 and its dependent claims patentably distinguish over the cited reference.

Therefore, reconsideration of the rejection is respectfully requested.

Respectfully submitted,

Date: July 27, 2005

Timothy N. Trop, Reg. No. 28,994
TROP, PRUNER & HU, P.C.
8554 Katy Freeway, Ste. 100
Houston, TX 77024
713/468-8880 [Phone]
713/468-8883 [Fax]

Attorneys for Intel Corporation